NEAMS Workbench 1.0 Beta Status

Robert A. Lefebvre, Adam B. Thompson, Brandon R. Langley, Bradley T. Rearden

Oak Ridge National Laboratory
1 Bethel Valley Road, Oak Ridge, Tennessee 37831 USA
lefebvrera@ornl.gov

INTRODUCTION

The NEAMS Workbench is a new initiative that will facilitate the transition from conventional tools to high-fidelity tools by providing a common user interface for model creation, review, execution, output review, and visualization for integrated codes.[1] The NEAMS Workbench has integrated Analysis Sequence Processors [2] that support specific application input and common input formats. The common user input format can include engineering-scale specifications that are expanded into application-specific input requirements through the use of customizable templates. Additionally, the common user input processor can be used to provide an enhanced alternative application input format that is easier to understand. A 1.0 beta release is available, and efforts to open source the NEAMS Workbench are under way. Expansion of the integrated codes and application templates available in the Workbench will broaden the NEAMS user community and will facilitate system analysis and design. Current and planned capabilities of the NEAMS Workbench are summarized here.

WORKBENCH OVERVIEW

The NEAMS Workbench will enable analysts to select the fidelity of each type of physics for use in the simulation, choosing from a variety of tools integrated from many projects and code teams. A conceptual design of tools that could be integrated for advanced reactor analysis is shown in Fig. 1. The current production tools with advanced components supported by the NEAMS program such as those from the Argonne Reactor Codes (ARC) [3] and the SCALE Code System [4] are shown in light gray. MCNP [5] and other production tools are shown in dark grey, tools from the NEAMS ToolKit are shown in maroon, and tools from the Consortium for Advanced Simulation of Light Water Reactors [6] are shown in black. With the Workbench, it will be possible to use the same fundamental engineering input data to create code-specific input models for each tool, enabling analysts to learn more about specific phenomena by performing reference high-fidelity analyses to confirm approximations or assumptions in fast-running lower-order design calculations.

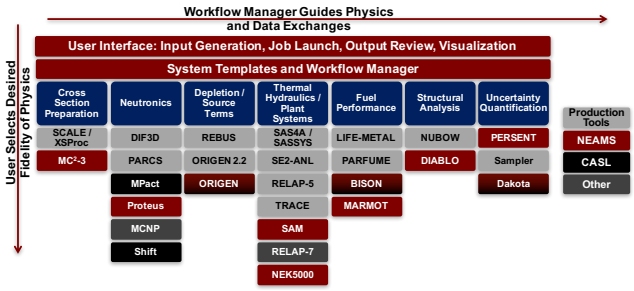

Fig. 1 Conceptual design of tools integrated in the NEAMS Workbench for advanced reactor analysis.

The NEAMS Workbench is leveraging the Fulcrum user interface, which was developed for the 2016 release of SCALE 6.2. Fulcrum introduced several new concepts as an integrated user interface for nuclear analysis. It builds on decades of user experience from thousands of users and replaces eight user interfaces from the 2011 release of SCALE 6.1. Fulcrum is a cross-platform graphical user interface designed to create, edit, validate, and visualize input, output, and data files. Fulcrum directly connects the user with the text form of the input file while providing inline features to assist with building correct inputs. Fulcrum provides input editing and navigation, interactive geometry visualization, job execution, overlay of mesh results within a geometry view, and plotting of data from file formats. An error checker interactively identifies input errors such as data entry omissions or duplications for all supported codes. The input validation engine identifies allowed data ranges and interdependencies in the input and reports inconsistencies to the user. The layout of panels in Fulcrum is highly configurable to accommodate the preferences of users, as shown in Fig. 2.

Notice: This manuscript has been authored by UT-Battelle, LLC, under contract DE-AC05-00OR22725 with the US Department of Energy (DOE). The US government retains and the publisher, by accepting the article for publication, acknowledges that the US government retains a nonexclusive, paid-up, irrevocable, worldwide license to publish or reproduce the published form of this manuscript, or allow others to do so, for US government purposes. DOE will provide public access to these results of federally sponsored research in accordance with the DOE Public Access Plan (http://energy.gov/downloads/doe-public-access-plan).

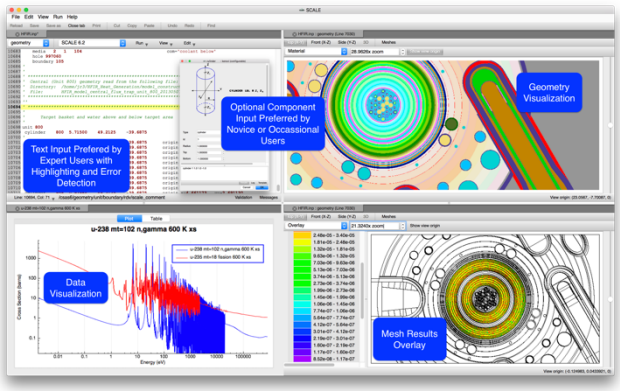

Fig. 2. The NEAMS Workbench leverages the Fulcrum user interface from SCALE.

The NEAMS Workbench extends Fulcrum to be application agnostic with components to facilitate integration of analysis sequence input processing, execution, and data visualization.

WORKBENCH ANALYSIS SEQUENCE PROCESSOR

The fundamental function of the NEAMS Workbench is the interaction of the user with an application input file. The Analysis Sequence Processor provides text file input processor capabilities that allow efficient lexical, parse, and validation processing. These capabilities provide Workbench with the algorithms and data structures for efficiently acquiring, analyzing, and presenting input data to the user. Additionally, a hierarchical template engine facilitates flexible text creation. The validation engine uses 19 input schema rules to allow describing input field restrictions on type, value, occurrence, and relation. Examples include the following.

- Type is an integer
- Specific value set or range. Value is a, b, c, or positive, etc.
- Aggregate value for specific input context. Sum of value.
- Existence/Uniqueness of value in relative input value sets. An identifier reference must exist in specific context elsewhere in the input.

Higher-order input validation (combinatorial geometry operation validation, binary data introspection, recursive relationships) is not supported. The Workbench Analysis Sequence Processor manual [2] provides input processing, creation, autocomplete, and validation capabilities to the NEAMS Workbench.

RUNTIME ENVIRONMENT

After careful consideration, the Python scripting language was chosen for its combination of cross-platform support, object-oriented design, code readability, and power. The runtime is an object-oriented polymorphic design allowing reuse and extension. A generic base class provides the complete interface to communicate command line options and usage and to facilitate job setup, execution, and finalization of an application's input execution.

Some applications have no runtime environment, requiring the user to manually conduct all steps associated with running the applications. For example, the user might be required to copy the problem input file into a specific location with a specific file name, input, and then invoke the application, thus producing temporary files and output file(s). This is error prone and should not be important to the user or to Workbench as this logic is application specific. The solution is to contribute a new runtime script—new_app.py—to the runtime repository, where the new_app.py overrides the setup, execution, or finalization logic as needed to fulfill the runtime interface. The setup logic might create a working directory, TMPDIR, and then copy the problem.inp into the TMPDIR as TMPDIR/input. The execution might invoke the application executable, passing application messages back to the calling application (command console, Workbench, etc.). The finalize logic might (1) combine the output files located in TMPDIR into logical order, (2) copy the output back into problem.out, residing next to problem.inp, and (3) delete the TMPDIR to clean up after itself.

Whatever the specific application logic is, an application.py script will be available to allow consistent invocation and potentially provide great convenience for the application's typical command line user. As the runtime environment matures and additional features are added (queuing system, etc.) for the base class, all incorporated runtimes will benefit.

If an application-specific runtime is too time-consuming to add, a generic Workbench runtime is available to naively execute an application for a given input. Application execution is currently limited to local execution. Cluster job scheduling is in development.

VISUALIZATION

The NEAMS Workbench provides the user 2D data and geometry visualization, as depicted in Fig. 2, and has integrated the VisIt [7] visualization toolkit. Integration of Kitware's ParaView visualization tool [8] is in the development plan. In addition to interactive 2D data and geometry plots, a 3D geometry renderer with surface opacity and culling controls provides enhanced perspective. The VisIt integration provides visualization support for many existing mesh and data formats. A NEAMS Workbench Processor capability facilitates text output data extraction and interactive 2D plot creation via a familiar Excel spreadsheet like cell-reference mechanism. A new processor file can be added at any time, providing Workbench users a means by which text data visualization can easily be customized or extended. The capabilities are depicted in Fig. 3.

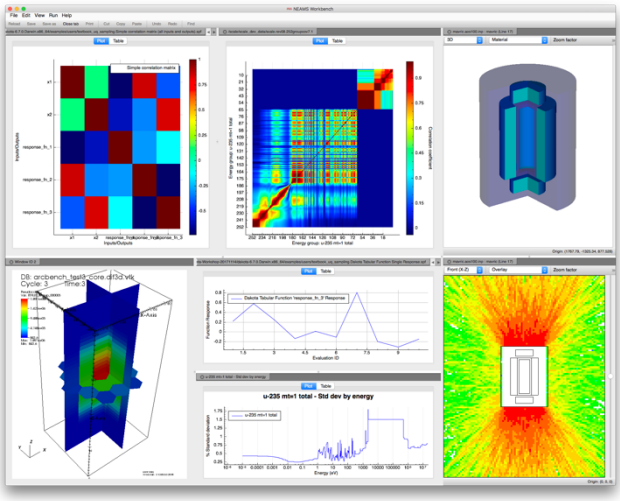

Fig. 3. Visualization capabilities.

From top left to bottom right, Fig. 3 shows the following.

1) Processor-created covariance plot
2) Nuclear data binary file–extracted covariance data plot
3) 3D view of a spent-fuel shipping cask with opacity and surface culling
4) Reactor flux map rendered by VisIt
5) Processor-created line plot of a response function by sample
6) Nuclear data binary file-extracted standard deviation by energy step plot
7) Spent-fuel shipping cask with dose-response Cartesian mesh data overlaid

The 2D geometry cross section view supports Cartesian and cylindrical mesh data overlay.

INTEGRATED TOOLS

Integrated tools minimally require the following.

1) Input processing integration to facilitate input creation, editing, navigation, and validation
2) Application runtime integration via generic or specific runtime script
3) Data visualization via supported data formats or Workbench Processor files

The tools that are currently integrated are as follows.

- MOOSE[9] BISON[10] fuel performance
- PyARC interface for ARC (MCC-3, DIF3D, REBUS, PERSENT) [11]
- Dakota UQ and Optimization [12]
- SCALE Code System

The MOOSE BISON fuel performance application is integrated at the level of the MOOSE framework. This means all MOOSE-based applications are also supported in the NEAMS Workbench, although BISON is the only application currently tested. Further integration of Dakota is under way to assist users in coupling Dakota to their analysis application.

The VisIt visualization tool is also integrated within the NEAMS Workbench program, providing convenient access to VisIt visualization and result analysis capabilities.

AVAILABILITY

The NEAMS Workbench beta 1.0 is currently available to any interested users and developers. An open source version is planned for the near future, but because of the origins of the Fulcrum user interface and its integrated development with SCALE, a SCALE license is currently required to request the beta version of the NEAMS Workbench.

Instructions for acquiring the NEAMS Workbench 1.0 Beta release are as follows:

1. Acquire RSICC License for the SCALE Code System https://rsicc.ornl.gov/codes/ccc/ccc8/ccc-834.html
2. Email nwb-help@ornl.gov; request NEAMS Workbench
    a. Include your SCALE RSICC license confirmation email
    b. Include the operating system(s) for which you would like prebuilt binary executables
3. Receive a response from nwb-help@ornl.gov
    a. Download the NEAMS Workbench deployment
    b. Follow the installation instructions

CONCLUSION

The NEAMS Workbench is a new initiative that will facilitate the transition from conventional tools to high-fidelity tools by providing a common user interface for model creation, review, execution, output review, and visualization for integrated codes. The continued integration of Dakota for UQ and Optimization, ARC via new innovative PyARC interface, SCALE via its Fulcrum heritage, MOOSE-BISON fuel performance, and VisIt visualization toolkit makes NEAMS Workbench 1.0 beta an impactful tool with significant potential. The integration of MCNP6, and additional NEAMS tools, such as NEK5000, along with runtime enhancements of job scheduling, and workflow

capabilities will provide a solid platform for users to accelerate and enhance their analysis workflows.

REFERENCES

1. B. T. REARDEN, R.A. LEFEBVRE, A. B. THOMPSON, B. R. LANGLEY, and N. E. STAUFF, "Introduction to the Nuclear Energy Advanced Modeling and Simulation Workbench," *M&C 2017 – International Conference on Mathematics & Computational Methods Applied to Nuclear Science and Engineering*, Jeju, Korea, April 16–20, 2017.
2. R. A. LEFEBVRE, B. R. LANGLEY, and J. P. LEFEBVRE, "Workbench Analysis Sequence Processor," ORNL/TM-2017/619, UT-Battelle, LLC, Oak Ridge National Laboratory (2017).
3. *ARC 11.0: Code System for Analysis of Nuclear Reactors*, Argonne National Laboratory (2014). Available from Available from Radiation Safety Information Computational Center as CCC-824.
4. B. T. REARDEN and M. A. JESSEE, eds., "SCALE Code System," ORNL/TM-2005/39, Version 6.2, UT-Battelle, LLC, Oak Ridge National Laboratory (2016). Available from Radiation Safety Information Computational Center as CCC-834.
5. D. B. PELOWITZ, ed., *MCNP6 User's Manual, Version 1.0,* Los Alamos National Laboratory report LA-CP-13-00634, Rev. 0 (May 2013). Available from Radiation Safety Information Computational Center as CCC-810.
6. J. A. TURNER, K. CLARNO, M. SIEGER, R. BARTLETT, B. COLLINS, R. PAWLOWSKI, R. SCHMIDT, and R. SUMMERS, "The Virtual Environment for Reactor Applications (VERA): Design and Architecture," *J. Comput. Phys.,* **326**, 544–568 (2016).
7. LLNL: VisIT Visualization Tool (2002–2016). https://wci.llnl.gov/codes/visit
8. Kitware: ParaView (2002–2016). http://www.paraview.org
9. D. GASTON, C. NEWMAN, G. HANSEN, and D. LEBRUN - GRANDIE´, "MOOSE: A parallel computational framework for coupled systems of nonlinear equations," *Nucl. Eng. Des.*, **239**, 1768–1778 (2009).
10. R. L. WILLIAMSON, J. D. HALES, S. R. NOVASCONE, M. R. TONKS, D. R. GASTON, C. J. PERMANN, D. ANDERS, and R. C. MARTINEAU, "Multidimensional multiphysics simulation of nuclear fuel behavior," *J. Nucl. Mater*, **423** 149–163 (2012).
11. N. STAUFF, N. GAUGHAN, and T. KIM, "ARC integration into the NEAMS Workbench," ANL/NE-17/31, September 30, 2017.
12. ADAMS, B.M., BAUMAN, L.E., BOHNHOFF, W.J., DALBEY, K.R., EBEIDA, M.S., EDDY, J.P., ELDRED, M.S., HOUGH, P.D., HU, K.T., JAKEMAN, J.D., STEPHENS, J.A., SWILER, L.P., VIGIL, D.M., AND WILDEY, T.M., "Dakota, A Multilevel Parallel Object-Oriented Framework for Design Optimization, Parameter Estimation, Uncertainty Quantification, and Sensitivity Analysis: Version 6.0 User's Manual," Sandia Technical Report SAND2014-4633, July 2014. Updated November 2015 (Version 6.3).